DOCUMENT RESUME

ED 420 725 TM 028 447

AUTHOR Jiang, Hai; Tang, K. Linda
TITLE New Method of Calibrating IRT Models.
PUB DATE 1998-04-00
NOTE 17p.; Paper presented at the Annual Meeting of the National Council on Measurement in Education (San Diego, CA, April 14-16, 1998).
PUB TYPE Reports - Evaluative (142) -- Speeches/Meeting Papers (150)
EDRS PRICE MF01/PC01 Plus Postage.
DESCRIPTORS Algorithms; *Item Response Theory; *Mathematical Models; Simulation; *Test Items
IDENTIFIERS *Calibration; *Optimization; Three Parameter Model

ABSTRACT

This discussion of new methods for calibrating item response theory (IRT) models looks into new optimization procedures, such as the Genetic Algorithm (GA) to improve on the use of the Newton-Raphson procedure. The advantages of using a global optimization procedure like GA is that this kind of procedure is not easily affected by local optima and saddle points. Because these procedures do not use gradient information, they can be implemented easily to higher dimensional data, even though they converge more slowly than the Newton-Raphson approach. However, the two approaches can be combined to exploit the advantages of both. That is, GA can be used to find a suitable starting point close to the global optima, and then Newton-Raphson can be used to speed up the convergence. The focus in this paper is on calibrating the unidimensional three-parameter logistic model (3PL) because that is the model most widely used in large-scale standardized tests. Using recent 3PL model estimates from recent Test of English as a Foreign Language administrations to generate examinee responses, the effectiveness of the new method is demonstrated using simulated data. How to implement the new methods with multidimensional data is discussed. (Contains 3 tables, 2 figures, and 10 references.) (SLD)

****
* Reproductions supplied by EDRS are the best that can be made *
* from the original document. *
****

ED 420 725

New Method of Calibrating IRT Models

Hai Jiang and K. Linda Tang

Educational Testing Service

March 20, 1998

Hai Jiang

U.S. DEPARTMENT OF EDUCATION
Office of Educational Research and Improvement
EDUCATIONAL RESOURCES INFORMATION CENTER (ERIC)
This document has been reproduced as received from the person or organization originating it.
Minor changes have been made to improve reproduction quality.
Points of view or opinions stated in this document do not necessarily represent official OERI position or policy.

Paper to be presented at the 1998 NCME annual meeting, San Diego, CA, April 15, 1998.

TM028447

Introduction

For simplicity, we assume the response data are dichotomous. That is, the responses are either right or wrong, coded as 1 or 0, respectively. We consider calibrating the multidimensional logistic model (the unidimensional three parameter logistic (3PL) model being its special case). The common approach to calibrating this class of models is to use the marginal maximum likelihood estimation approach by Bock and Aitkin (1981). That is, we maximize the marginal likelihood function, the likelihood function integrated over the latent ability distribution. To achieve this, the so-called EM algorithm by Dempster, Laird, and Rubin (1977) needs to be used because of the difficulty of directly maximizing the marginal likelihood function. In the M (Maximization) step of the EM algorithm, optimization procedures such as Newton-Raphson are used, resulting in BILOG (Mislevy and Bock, 1982) for calibrating the unidimensional 3PL model and later TESTFACT (Bock, Gibbons, and Muraki, 1988) for calibrating multidimensional logistic model. In this paper, we will be looking into new optimization procedures such as Genetic Algorithm (GA) to improve upon the use of Newton-Raphson. The advantages of using global optimization procedures such as GA are that this kind procedure won't be easily fooled by local optima and saddle points. And because they don't use gradient information, they can be easily implemented to higher dimensional data. Yet also because of this, they converge slower than Newton-Raphson. However, we can combine the two approaches to fully exploit their respective advantages. That is, we can use GA to find a suitable starting point that is close enough to the global optima, and then use Newton-Raphson to speed up the convergence. We will concern mostly on calibrating the unidimensional 3PL model in this paper because that model is by far the most widely used one in large-scale standardized tests. Using unidimensional 3PL model estimates from recent TOEFL administrations to generate examinee responses, we will show the effectiveness of our new method using these simulated data. Finally we will discuss briefly on how to implement our new method to multidimensional data.

Method

The unidimensional 3PL model

Assume there are a total of I items in the test. Under the unidimensional 3PL model, the probability of answering item i correctly given that the examinee has ability θ is

$$P_i(\theta;\beta_i) = c_i + \frac{1-c_i}{1+\exp(-1.701a_i(\theta-b_i))}$$

where a_i is the discrimination, b_i is the difficulty, c_i is the guessing for item i, and $\beta_i = (a_i, b_i, c_i)$ is the vector of item parameters.

Assume θ is distributed as $N(\mu, \sigma^2)$, the normal distribution with mean μ and variance σ^2. Appropriate transformations on θ as well as on the item parameters will make θ distributed as $N(0, 1)$, the standard normal distribution, and give the same likelihood function. Thus, the density of θ is assumed to be

$$\pi(\theta) = \frac{1}{\sqrt{2\pi}}\exp(-\frac{\theta^2}{2})$$

Using the local independence of the examinee responses given ability θ, and denote the totality of item parameters as $\mathbf{B} = (\beta_1, \ldots, \beta_n)$, the marginal likelihood of the response matrix $\mathbf{Y}$ is given by

$$L(\mathbf{Y};\mathbf{B}) = \prod_s \int P(\mathbf{Y}_s|\theta;\mathbf{B})\pi(\theta)d\theta$$

where

$$P(\mathbf{Y}_s|\theta;\mathbf{B}) = \prod_{i=1}^{n} P(Y_{si}|\theta;\beta_i) = \prod_{i=1}^{n} P_i(\theta;\beta_i)^{Y_{si}}(1-P_i(\theta;\beta_i))^{1-Y_{si}}$$

Taking logarithm, the log likelihood is

$$\ln L(\mathbf{Y};\mathbf{B}) = \sum_s \ln \int P(\mathbf{Y}_s|\theta;\mathbf{B})\pi(\theta)d\theta$$

The EM algorithm

Since directly maximizing $\ln L(\mathbf{Y};\mathbf{B})$ over $\mathbf{B}$ is infeasible, we use the EM algorithm.

The EM algorithm, as its name suggests, is divided into two steps: the E (Expectation) step, and the M (Maximization) step. Cyclical application of the E step and the M step continues till a certain convergence criterion is met.

In the E step, the conditional expectation of log likelihood of complete data given the incomplete data and current parameter estimates is computed. In calibrating the unidimensional 3PL model, the incomplete data is the observed response matrix $\mathbf{Y}$ and the complete data is the responses plus the examinee latent ability vector θ. So in the E step, the following quantity is computed

$$Q(\mathbf{B};\mathbf{B}') = E[\ln L(\mathbf{B}|\mathbf{Y},\theta)\,|\,\mathbf{Y};\mathbf{B}']$$

where the expectation is taken with respect to θ. Here $\mathbf{B}'$ is the parameter estimates resulted from the M step in the previous iteration. Here and below we follow the

DATE: 05/13/98

ERIC/TME Clearinghouse Report

Document Cover Sheet

TM #: 028447

Title: NEW METHOD OF CALIBRATING IRT MODELS

Number of Pages: 20

Publication Date: 04/14/98

Document Level: 1

Notes:

standard notation in the literature of EM algorithm. It is understood that the Q functions in the E and the M steps depend on the observed response matrix $\mathbf{Y}$.

For the unidimensional 3PL model the following decomposition holds

$$Q(\mathbf{B};\mathbf{B}') = \sum_{i=1}^{I} Q_i(\beta_i;\mathbf{B}') + \text{const.}$$

where $Q_i(\beta_i;\mathrm{B}') = \sum_k \int \ln P(Y_{ik}|\theta;\beta_i)\xi(\theta|\mathbf{Y}_k;\mathrm{B}')\pi(\theta)d\theta$ involves only the item parameter β_i for item i, and $\xi(\theta|\mathbf{Y}_k;\mathrm{B}') = P(\mathbf{Y}_k|\theta;\mathrm{B}') / \int P(\mathbf{Y}_k|\theta;\mathrm{B}')\pi(\theta)d\theta$ is the posterior density.

In the M step, $Q(\mathbf{B};\mathbf{B}')$ is maximized over the parameters $\mathbf{B}$ for given $\mathbf{B}'$ and $\mathbf{Y}$. Because of the decomposition in the E step, we can separately maximize each $Q_i(\beta_i;\mathbf{B}')$ over β_i.

Let $A(\theta) = \sum_k \xi(\theta|\mathbf{Y}_k;\mathrm{B}')$, and $R_i(\theta) = \sum_k Y_{ik}\xi(\theta|\mathbf{Y}_k;\mathrm{B}')$, we get

$$Q_i(\beta_i;\mathrm{B}') = \int [\ln P_i(\theta;\beta_i)R_i(\theta) + \ln(1-P_i(\theta;\beta_i))(A(\theta)-R_i(\theta))]\pi(\theta)d\theta$$

Unfortunately, no closed-form solution exists for maximizing $Q_i(\beta_i;\mathbf{B}')$ over β_i. We could use optimization procedures such as Newton-Raphson. However, procedures using gradient information such as Newton-Raphson behave only locally. That is, they only work if their starting points are close enough to the global optima. Otherwise, they could be easily trapped to a local optima or even saddle points. To avoid this, we use Genetic Algorithm (Michalewicz, 1994) to first find a suitable starting point for Newton-Raphson. Once we are close enough to the true global optima, we use Newton-Raphson to speed up the convergence.

Genetic Algorithm

Any optimization task can be thought of as a search through a space of potential solutions. Genetic Algorithm (GA) is a stochastic algorithm whose search method emulates the natural phenomena of genetic inheritance and Darwinian strife for survival. A GA maintains a population of individuals $P(t) = \{x_1^t, \ldots, x_n^t\}$ for use in iteration t. Each individual is a vector and represents a potential solution to the problem at hand (i.e., a potential optimizer of the problem). Each solution x_i^t is evaluated to give some measure of "fitness". Then, as a result of iteration t a new population $P(t+1)$ for use in iteration $t+1$ is formed by selecting the more "fit" individuals (select step). Some members of this new population undergo transformations (alter step) by means of "genetic" operators to form new potential solutions. There are unary transformations (mutation type), which

create new individuals by a small change in a single individual, and higher order transformations (crossover type), which create new individuals by combining segments from several (two or more) individuals. After several generations the program converges with the goal being that the best individual in this final generation represents a near-optimum solution.

A Genetic Algorithm for a particular problem must have the following components:

- a representation for potential solutions to the problem,
- a way to create an initial population of potential solutions,
- an evaluation function that plays the role of the environment, rating solutions in terms of their "fitness",
- genetic operators that alter the composition of offspring,
- values for various parameters that the Genetic Algorithm uses (population size, probabilities of applying genetic operators, etc.)

Implementation of a Genetic Algorithm

As an example, let us consider item 1. Dropping the index of item for the parameters, the item parameters to be estimated are $\beta = (a,b,c)^T$. At the current M step, we are trying to maximize $Q_1(\beta;B')$ over β.

A Genetic Algorithm has the following elements:

1. population of solutions.

As we have mentioned above, an important property of Genetic Algorithm is that it maintains a population of potential solutions while conventional search methods such as Newton-Raphson process a single point of search space.

For the maximization problem here, a population of potential solutions is a set of J vectors $\beta_1,\ldots,\beta_J$ where $\beta_j = (a_j,b_j,c_j)^T$

2. initialization of the population.

We first replicate the maximizer of β from the previous EM cycle, $\tilde{\beta}$, J times to get a set of J $\tilde{\beta}$'s. After the genetic operations defined later, this set of vectors gives the initial population $P(0)$, where they are vastly different from each other. Intuitively the maximizer of β from the previous EM cycle is close to the "true" value of β. After the genetic operations, the chances are some of them are little to unchanged, yet we have

added the variability to the initial population, which in a way is an essence of Genetic Algorithm.

3. evaluation function.

An obvious choice is the objective function to be maximized, in our case $Q_1(\beta;B')$. In theory, any monotone transformation of the objective function can be used as the evaluation function, so the choice is determined by the ease of computation of a specific transformation. The value of the evaluation function at a possible solution gives a measure of "fitness" of that solution.

After the evaluation function has been chosen, the selection procedure needs to be determined. Theoretically, any selection procedure that has the probability of a possible solution being chosen proportional to the value of a monotone transformation of the evaluation function at the solution is allowed. Our selection procedure uses the rank of the value of evaluation function at a possible solution as a basis to select the more fit solutions.

For each possible solution β_j, we first compute $Q_1(\beta_j;B')$ and rank β_j in ascending order according to its Q_1 value. Suppose the ranks are $r_1,\ldots,r_J$, then the probability of β_j being selected is

$$P(\beta_j)=\frac{r_j}{\sum_{j=1}^{J} r_j}=\frac{2r_j}{J(J+1)}$$

The higher the rank, the more likely a possible solution gets selected.

The advantage of using ranks as basis for selection is the scale of selection probability is comparable for all the possible solutions. If the selection probability is based on values of evaluation function at possible solutions, it may happen that some of the solutions give so small a value of evaluation function that they seldom get selected. Consequently, the possibility of have variability by selection cannot be well achieved.

4. genetic operators.

The genetic operators for our problem are mutation and crossover.

• **crossover**

Even though we say crossover is a binary transformation, it differs from the conventional transformation in that a pair of vectors are transformed into a new pair of vectors instead of a single vector.

To perform the crossover operation, we randomly select $p_c \cdot J$ solution vectors for crossover, where p_c is the probability for the crossover operation. For the solution vectors selected, first we pair them up, then each pair generates a new pair by the scheme below:

If $s_a^t = (v_1, \ldots, v_m)^T$ and $s_a^t = (w_1, \ldots, w_m)^T$ are crossed after the k-th position, the resulting offspring are:

$$s_a^{t+1} = (v_1, \ldots, v_k, w_{k+1}, \ldots, w_m)^T \text{ and } s_a^{t+1} = (w_1, \ldots, w_k, v_{k+1}, \ldots, v_m)^T$$

Here the position of crossover k is chosen randomly. After the crossover, the original pair is discarded.

• **mutation**

Mutation is a unary transformation that generates a new solution vector from a chosen solution vector. To perform the mutation operation, we randomly select $p_m \cdot J$ solutions for mutation, where p_m is the probability of mutation. The vectors chosen are then mutated accordingly using the scheme below:

If $s_a^t = (v_1, \ldots, v_m)^T$ is a solution vector chosen, the resulting offspring is a vector $s_a^{t+1} = (v_1, \ldots, v_m)^T$, where v_k is v_k plus a random noise ε_k distributed as $N(0, \sigma_k)$.

5. values for the parameters of Genetic Algorithm.

The population size can be anywhere from tens to thousands, and there is a tradeoff between accuracy and efficiency. We usually use 100 as the population size.

When initializing the population, we set the probabilities of applying genetic operators to be relatively large to get more variability because we start with replications of the maximizer from the previous EM cycle. So for example, on initialization, the probability setting can be $p_m = 0.7$ and $p_c = 0.5$. After initialization, for each generation of evolution, the probability setting can be $p_m = 0.5$ and $p_c = 0.3$.

The maximum generation number is set at 100.

So each generation of the evolution consists of a cycle of mutation, crossover, and selection, and after each generation we always keep the best solution vector, which is β_0.

Using our Genetic Algorithm, the maximizer $\hat{\beta}$ of $Q_1(\beta; \mathrm{B}')$ is given by the best solution vector from the generation of evolution at the stopping time.

Looping over the E and M steps, we get an EM sequence of item parameter estimates. We can stop the EM process either after a fixed number of iterations, or after the variation of log likelihood $\ln L(\mathbf{Y};B)$ using the maximizer of B from the previous several EM cycles has been small enough.

The item parameter estimates B can then be input as the starting point for a BILOG like estimation procedure.

Simulation study and results

Simulation settings

We use item parameter estimates from Section 2 (Structure and Written Expression) of three recent Test of English as a Foreign Language (TOEFL) administrations to simulate examinee responses. This section has 38 operational items. Assuming a unidimensional 3PL model, the item parameter estimates were obtained through calibration of examinee responses using BILOG. The number of examinees in our simulation study is either 500 or 1000. The sample size of 500 reflects the lower limit for calibrating the TOEFL operational forms, and the larger sample of 1000 is chosen to see how much an effect of sample size is to the calibration of the tests using our new method. For each sample size, the examinee ability distribution is assumed to be $N(0, 1)$. When using BILOG, the calibration parameters are set using empirical evidence as below:

- item control parameters are set at a=1.00, b=-0.50, and c=0.23,
- prior distribution for a is Log-Normal with Mean 0.00 and default S.D. 0.50,
- prior distribution for b is Normal with Mean -0.50 and default S.D. 2.00,
- prior distribution for c is Beta with Alpha 4.45 and Beta 12.55 (corresponding to a weight of 15.00 for subjects of low ability), and
- subject prior distribution is $N(0, 1)$.

To compare the calibration results, we compute the summary statistics of the item parameter estimates from different method and compare them with those of the true item parameters. Since different sets of item parameters can give almost identical item characteristic curves (ICCs), we also compute the root mean squared difference (RMSD) between the ICC from the item parameter estimates and the one from the true parameters using the formula below:

$$\text{ICC RMSD} = \left[\int (P_{i,est}(\theta) - P_{i,true}(\theta))^2 \pi(\theta) d\theta\right]^{\frac{1}{2}}$$

It is the ICCs rather than the item parameters themselves that are used in statistical analyses.

Since the test characteristic curve (TCC) is used in operations such as the IRT true score equating, in scaling, and in test analysis, we compute the RMSD between the TCC from the item parameter estimates and that from the true parameters using a formula similar to the one used in computing the RMSD between the ICCs. We also include overlay plots of TCCs using the true parameters and the parameter estimates from different methods to visually show how close they are throughout the ability range of our concern.

Simulation results

Tables 1 and 2 give the summary statistics of the item parameter estimates using different methods and those of the true parameters for sample sizes of 1000 and 500, respectively. Also shown in Tables 1 and 2 are the summary statistics of the ICC RMSDs between the item parameter estimates and the true parameters as well as the TCC RMSDs between the item parameter estimates and the true parameters.

Table 1. Comparison of results using different calibration methods

sample size=1000

	a		*b*		*c*		ICC RMSD		
	Mean	S.D.	Mean	S.D.	Mean	S.D.	Mean	S.D.	TCC RMSD
					Form A				
BILOG	1.035	0.317	-0.382	0.840	0.250	0.095	0.025	0.011	0.338
GA	1.091	0.379	-0.439	0.795	0.257	0.168	0.028	0.013	0.584
True	0.948	0.259	-0.443	0.960	0.239	0.127			
					Form B				
BILOG	1.061	0.292	-0.182	0.882	0.227	0.072	0.024	0.011	0.340
GA	1.072	0.349	-0.389	0.907	0.196	0.127	0.038	0.018	1.065
True	0.943	0.236	-0.283	0.993	0.198	0.099			
					Form C				
BILOG	1.066	0.247	-0.350	0.766	0.245	0.085	0.024	0.012	0.287
GA	1.113	0.311	-0.432	0.687	0.243	0.167	0.031	0.016	0.665
True	0.997	0.245	-0.409	0.830	0.229	0.110			

Table 2. Comparison of results using different calibration methods

sample size=500

	a		*b*		*c*		ICC RMSD		
	Mean	S.D.	Mean	S.D.	Mean	S.D.	Mean	S.D.	TCC RMSD
					Form A				
BILOG	1.045	0.359	-0.471	0.862	0.240	0.073	0.028	0.016	0.346
GA	1.093	0.448	-0.559	0.896	0.213	0.164	0.034	0.017	0.696
True	0.948	0.259	-0.443	0.960	0.239	0.127			
					Form B				
BILOG	1.069	0.367	-0.252	0.902	0.218	0.054	0.026	0.017	0.347
GA	1.067	0.392	-0.457	0.898	0.184	0.147	0.046	0.023	1.330
True	0.943	0.236	-0.283	0.993	0.198	0.099			
					Form C				
BILOG	1.076	0.328	-0.429	0.787	0.232	0.060	0.029	0.016	0.341
GA	1.135	0.407	-0.549	0.779	0.204	0.164	0.039	0.020	0.894
True	0.997	0.245	-0.409	0.830	0.229	0.110			

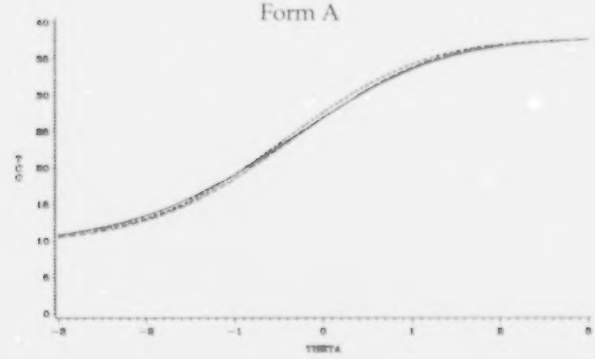
Li (1997) has shown that using GA alone gives comparable calibration results as those given by using BILOG. From Tables 1 and 2, it is clear that using GA alone gives acceptable calibration results for most of the statistical analysis purposes where the ICCs or the TCCs are used. Certainly it is not surprising to see that with a larger sample size the calibration results become more accurate.

Figures 1 and 2 below give the TCC overlay plots for the three forms in our study with Figure 1 for sample sizes of 1000 and Figure 2 for sample sizes of 500. There are three curves in each plot: the solid one is the TCC from the true item parameters, the long dashed one is the TCC from the item parameter estimates using BILOG, and the short dashed one is the TCC from the item parameter estimates using GA alone.

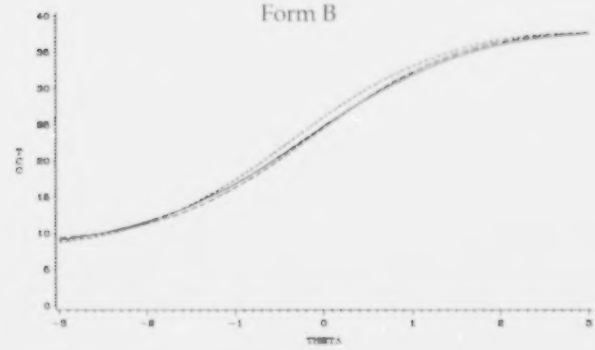
From Figures 1 and 2, it is clear that the TCCs from the item parameter estimates using BILOG and using GA alone are quite close to the TCC from the true parameters for each of the three forms and either sample size. Certainly it appears that the TCC from BILOG item parameter estimates is closer to the truth than the TCC from GA item parameter estimates for each form (especially Form B) and either sample size. It seems true also that the TCCs are more accurately estimated using a larger sample size.

Figure 1. TCC overlay plots
sample size=1000

Form A



Form B



Form C

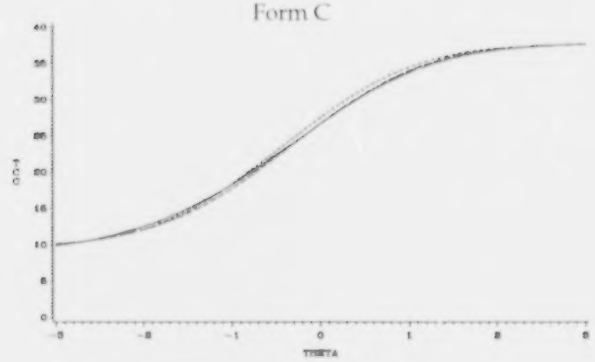
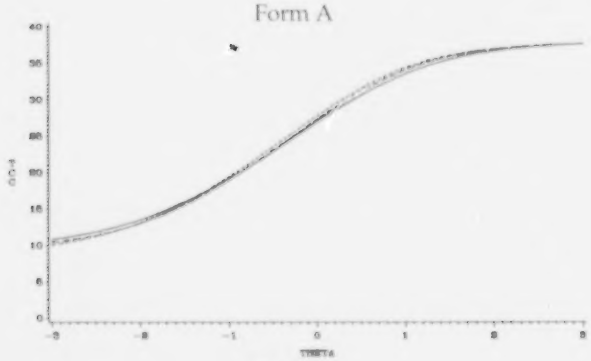
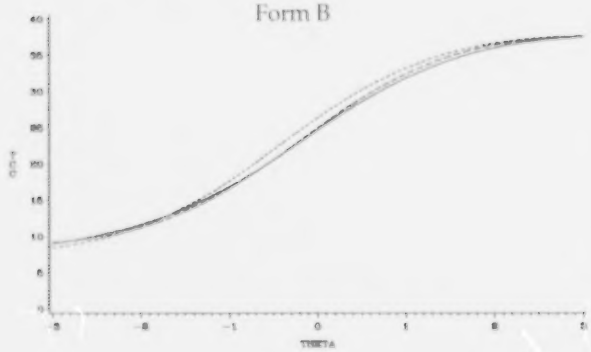


Figure 2. TCC overlay plots
sample size=500

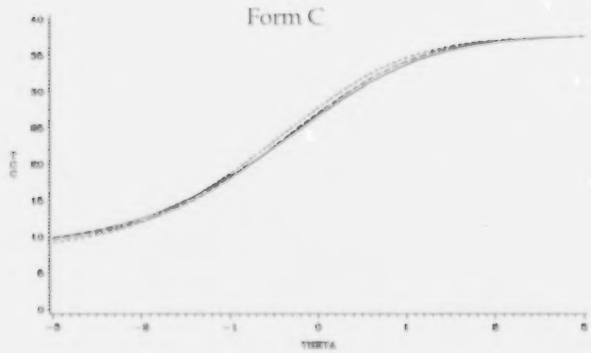
Form A



Form B



Form C



It turns out that using GA to get the starting points and then using BILOG gives identical calibration results as using BILOG which is no surprise. However, the number of EM cycles for BILOG is reduced and the increase in log likelihood is much smaller as if BILOG is only doing fine tuning when the starting points are determined by GA. Table 3 shows the number of EM cycles and the increase in log likelihood using GA then BILOG as compared to those using BILOG alone.

Table 3. Number of EM cycles and increase in log likelihood using different methods

	number of EM cycles	increase in log likelihood
	Form A, sample size=1000	
GA then BILOG	30	6.469
BILOG	36	2965.759
	Form A, sample size=500	
GA then BILOG	27	8.082
BILOG	34	1555.711
	Form B, sample size=1000	
GA then BILOG	38	8.055
BILOG	49	3614.708
	Form B, sample size=500	
GA then BILOG	40	0.770
BILOG	50	1870.096
	Form C, sample size=1000	
GA then BILOG	30	5.198
BILOG	39	2730.275
	Form C, sample size=500	
GA then BILOG	33	8.909
BILOG	37	1400.397

As can be seen from the above table, the number of EM cycles is reduced by at least 10% (and sometimes more than 20%). More important to note yet, the increase in the objective function of log likelihood is less than 10 (compared with the final values which are in the magnitude of 18000 for sample sizes of 1000 and 9000 for sample sizes of 500). Also from Table 3, the number of EM cycles and the increase in log likelihood are significantly larger for Form B for both sample sizes than for the other two forms. By examining the priors and the truth closely, we see that the priors and the control

parameters for b are not as close to the truth for Form B as for the other two forms. We believe consequently this causes the BILOG to take more EM cycles to converge.

Note

Since the GA in our study is used to provide the starting points for BILOG only, we have deliberately avoided using some of the fine tuning techniques available from the literature. Also, we have set a rather loose convergence criterion for the GA. Yet it is clear from our results above that using GA alone would give us quite satisfactory calibration results in terms of the ICCs and TCCs.

Discussion

Statistical models are used to summarize the information contained in a data in a simplified and realistic way. The parameter estimates of an underlying model for the data are used in many different areas. Accurate estimates of parameters are crucial to many operations of a testing program, such as in IRT true score equating, in scaling, in item and test analysis, and in research. Our new method is very promising in giving accurate calibration results more efficiently of the unidimensional 3PL model which is now widely used in many large-scale testing programs. Also as discussed below, the method can be easily generalized to calibrate multidimensional logistic model, which many researchers become more and more interested in using. As we mentioned earlier, a unique feature that distinguishes GA from other methods is that a GA performs a multi-directional search by maintaining a population of potential solutions and encourages information formation and exchange between these directions. Gas have been quite successfully applied to optimization problems like scheduling, adaptive control, cognitive modeling, optimal control problems, and database query optimization (Bennett, Ferris, and Ioannidis, 1991; DeJong, 1985; Goldberg, 1989; Michalewicz, Krawczyk, Kazemi, and Janikow, 1990). Since Gas are parallel in nature, with parallel computing becoming more and more practical, our new calibration method will for sure become more computing efficient.

Implementing the GA to multidimensional logistic model

Certainly the solution vectors need to be changed accordingly, and the genetic operators need to be changed to reflect the changes in the dimension of the solution vectors. However, it is obvious these changes are quite straightforward especially when compared with the changes needed in computing the gradient vectors and the Hessian matrix.

References

Bennett, K., Ferris, M.C., and Ioannidis, Y.E. (1991). A genetic algorithm for database query optimization. In Belew, R., and Booker, L. (Eds.), *Proceedings of the Fourth International Conference on Genetic Algorithms*. Los Altos, CA: Morgan Kaufmann Publishers.

Bock, R.D. and Aitkin, M. (1981). Marginal maximum likelihood estimation of item parameters: application of an EM algorithm. *Psychometrika, 46*, 443-59.

Bock, R.D., Gibbons, R., and Muraki, E. (1988). Fill-information item factor analysis. *Applied Psychological Measurement, 12*, 261-80.

DeJong, K.A. (1985). Genetic algorithms: a 10 year perspective. In Grefenstette, J.J. (Ed.), *Proceedings of the First International Conference on Genetic Algorithms*. Hillsdale, NJ: Lawrence Erlbaum Associates.

Dempster, A.P., Laird, N.M., and Rubin, D.B. (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, Series B, 39*, 1-38.

Goldberg, D.E. (1989). *Genetic algorithms in search, optimization, and machine learning*. Reading, MA: Addison-Wesley.

Li, H.H. (1997). Using genetic algorithms to estimate IRT item parameters. Paper presented at the 1997 Annual Meeting of Psychometric Society. Gatlinburg, TN.

Michalewicz, Z. (1994). *Genetic algorithms + data structures=evolution programs*. Berlin: Springer-Verlag.

Michalewicz, Z., Krawczyk, J., Kazemi, M., and Janikow, C. (1990). Genetic algorithms and optimal control problems. *Proceedings of the 29th IEEE conference on Decision and Control*. Honolulu, HI.

Mislevy, R., and Bock, R.D. (1982). *BILOG: item analysis and test scoring with binary logistic models*. Mooresville, IN: Scientific Software.